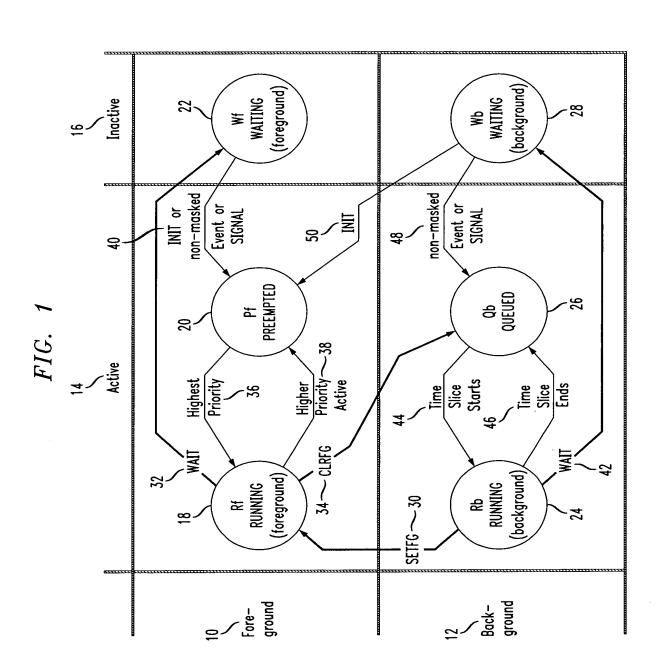DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

1/21

FIG. 1

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

2/21

FIG. 2

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

3/21

FIG. 3

status bits per context — 86

Event Status [7:0] — 94

bits =1 indicate asserted activation events

ACT — 92

=1 if context is active

control bits per context — 84

Event Mask [7:0] — 90

bits =1 enable corresponding activation events

FG — 88

=1 if context is in foreground

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
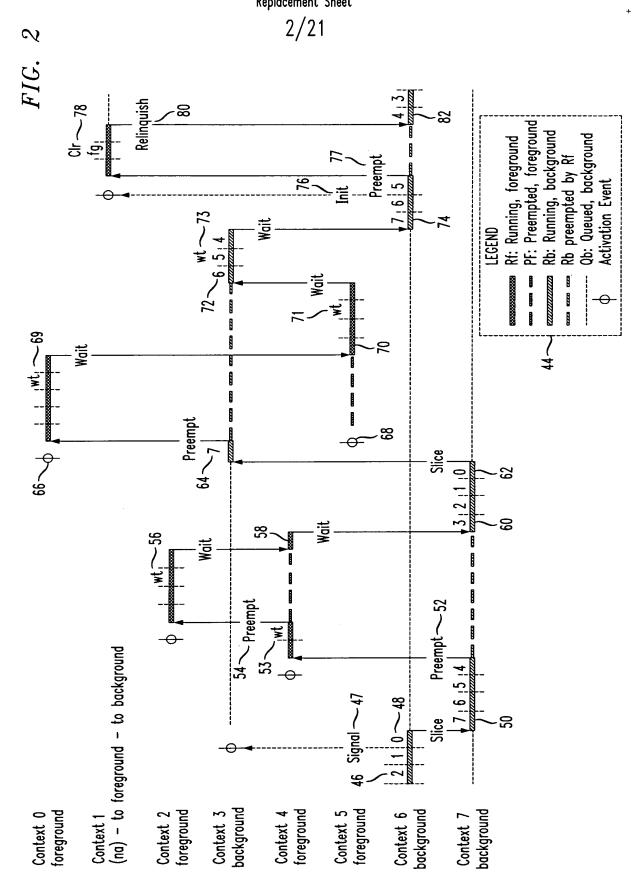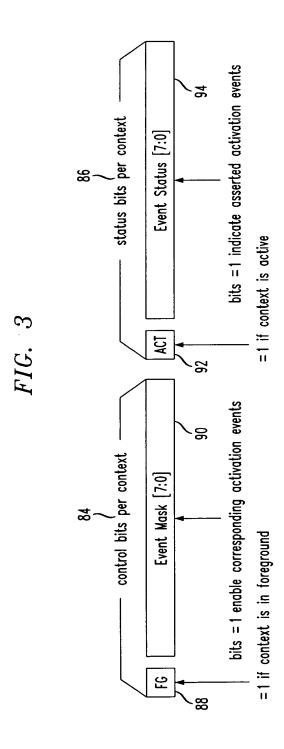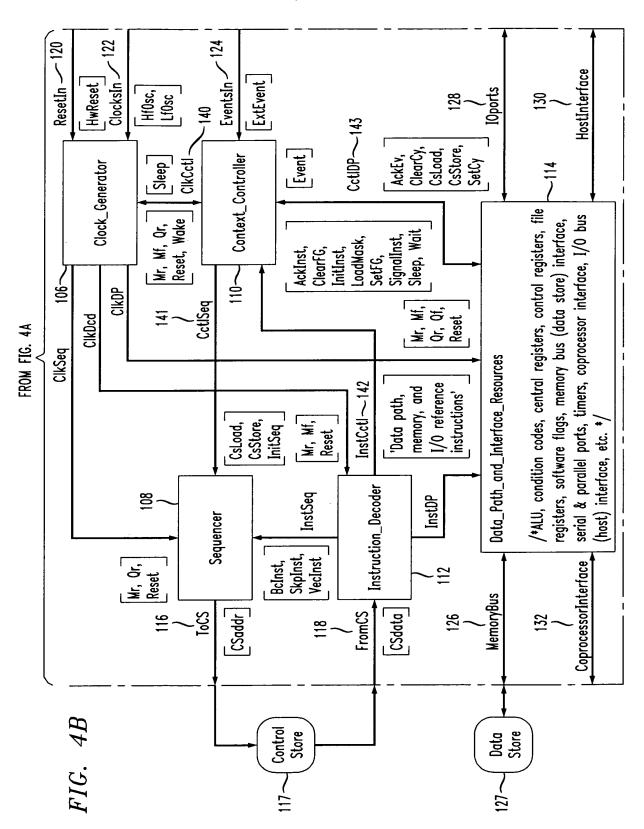Replacement Sheet

4/21

FIG. 4A

system IO_Controller ～100   1(1)

```
/* BitString8 & 16 based on Bit_String from Z.105 */
syntype Cgroup = Integer constants 0:3 endsyntype;
syntype Cond = Integer constants 0:31 endsyntype;
syntype CtxNum = Integer constants 0:7 endsyntype;
syntype EventNum = Integer constants 0:7 endsyntype ;
syntype InstAddr = Integer constants 0:65535 endsyntype;
syntype Instruction = BitString16 endsyntype ;
syntype Offset = Integer constants -128:127 endsyntype;
syntype Vbase = Integer constants 0:1023 endsyntype;
    /* Exported from Context_Controller */
remote asleep, csw, idle Boolean nodelay ;
remote context CtxNum nodelay ; /* running context */
remote events BitString8 nodelay ; /* C-group 01 */
remote nctx CtxNum nodelay ; /* next context */
remote mask BitString8 nodelay ; /* Event Mask reg */
    /* Exported from Data_Path_and_Interface_Resources */
remote slice Natural nodelay ; /* inst cycles per bg slice */
remote ien Boolean nodelay ; /* true for execute cycles */
```
～102

```
signal
    AckEv(CtxNum,EventNum),
    AckInst(EventNum),
    BcInst(Cond,Offset), ClearCy(CtxNum),
    ClearFG, CSaddr(InstAddr),
    CSdata(Instruction), CsLoad(CtxNum),
    CsStore(CtxNum),
    Event(CtxNum,EventNum),
    ExtEvent(CtxNum,EventNum), HfOsc,
    HwReset, InitInst(CtxNum),
    InitSeq(CtxNum), LfOsc,
    LoadMask(BitString8), Mr, Mf,
    Qr, Qf, Reset, SetCy(CtxNum),
    SetFG, SignalInst(CtxNum,EventNum),
    SkpInst(Cgroup,BitString8), Sleep,
    VecInst(Vbase), Wait, Wake ;
```
～104

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

5/21

FIG. 4B

FROM FIG. 4A

ResetIn — 120
[HwReset]
ClocksIn — 122
[HfOsc, LfOsc]
EventsIn — 124
[ExtEvent]

Clock_Generator 106

[Sleep] ClkCctl — 140

[Mr, Mf, Qr, Reset, Wake] 141

Context_Controller 110

[Event]

CctlDP — 143
[AckEv, ClearCy, CsLoad, CsStore, SetCy] 128 IOports

HostInterface 130

114

ClkSeq
ClkDcd
ClkDP
CctlSeq

[AckInst, ClearFG, InitInst, LoadMask, SetFG, SignalInst, Sleep, Wait]
[Mr, Mf, Qr, Qf, Reset]

InstCctl — 142

[CsLoad, CsStore, InitSeq]
[Mr, Mf, Reset]

['Data path, memory, and I/O reference instructions']

Sequencer 108

InstSeq

Instruction_Decoder 112

InstDP

Data_Path_and_Interface_Resources

/*ALU, condition codes, central registers, control registers, file registers, software flags, memory bus (data store) interface, serial & parallel ports, timers, coprocessor interface, I/O bus (host) interface, etc. */

[Mr, Qr, Reset] 116
ToCS [CSaddr]

[BcInst, SkpInst, VecInst] 118
FromCS [CSdata]

126 MemoryBus

132 CoprocessorInterface

Control Store 117

Data Store 127

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

6/21

FIG. 5

ClkCctl —140

block Context_Controller   [Sleep]   1(1)

110

156

ClkSyn

Mr,
Reset

154   158   124

ClkPri   AsyncEvents   EventsIn
                      150— Event_Synchronizer
                              (1,1)   [ExtEvent]

CsLoad,
CsStore,   160
InitSeq
141   PriSeq   Mr,
                                Mf, Qr,
CctlSeq   Reset,
                                Wake   SyncEvents

162   152   166

InstPri

142   Event_Prioritizer   [Event]
                        (1,1)
InstCctl

AckInst,
ClearFG,
InitInst,
LoadMask,   [Event]
SetFG,
SignalInst,
Sleep, Wait   164

PriDP

AckEv,
ClearCy,
CsLoad,
CsStore,
SetCy

CctlDP

143

FIG. 6   7/21

process Event_Synchronizer —150   1(1)

```
signal EndMark, ResetMark ;
signalset EndMark, ExtEvent,
  Mr, Reset, ResetMark ;

dcl c# CtxNum ;
dcl e# EventNum ;
```
200

/* This process synchronizes events originating outside of
the DISC core to the leading edge of Mr. The assertion of
each external event is indicated by an ExtEvent signal. The
parameters of the signal provide the target context and
event numbers.

ExtEvent signals are saved in the process' input queue
until receipt of signal Mr from the Clock_Generator block.
The end of the input queue is marked and all ExtEvent
signals queued ahead of the EndMark are copied to Event
signals and sent to the Event_Prioritizer process, where
they are handled at the next Mr. */
201

202

204 — Hold_Events

206 — Mr   ExtEvent —208   * —224

210 — EndMark to self   Reset —226

212 — Pass_Events   ResetMark to self —228

214 — ExtEvent (c#, e#)   EndMark —216   Flush_Events —230

218 — Event (c#, e#)   Hold_Events —220   ResetMark —232

222 — –   Hold_Events —234

DIEPSTRATEN 16-2-2
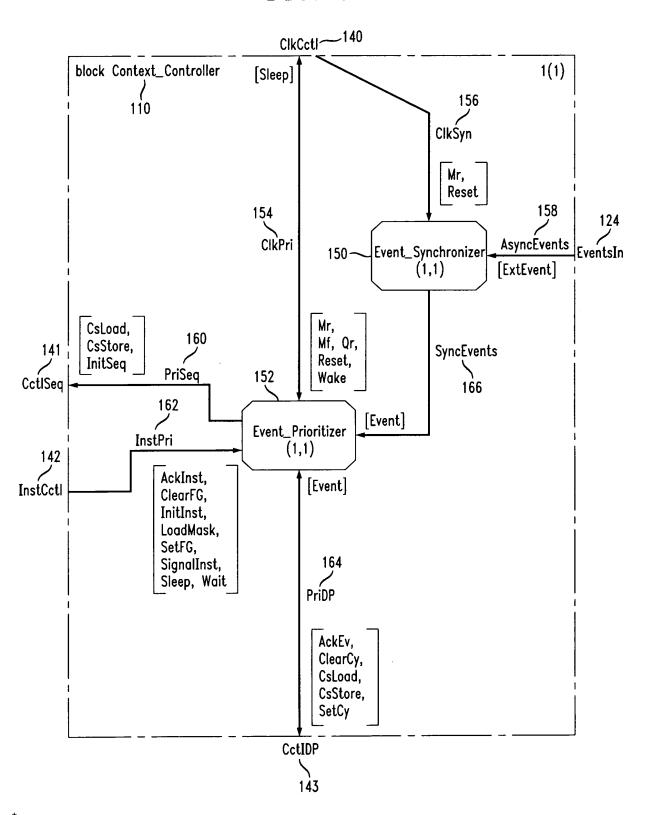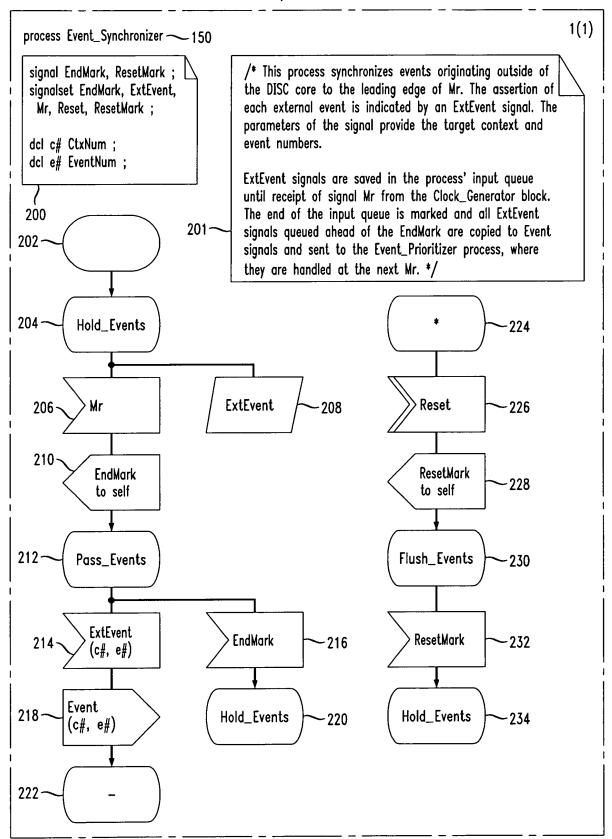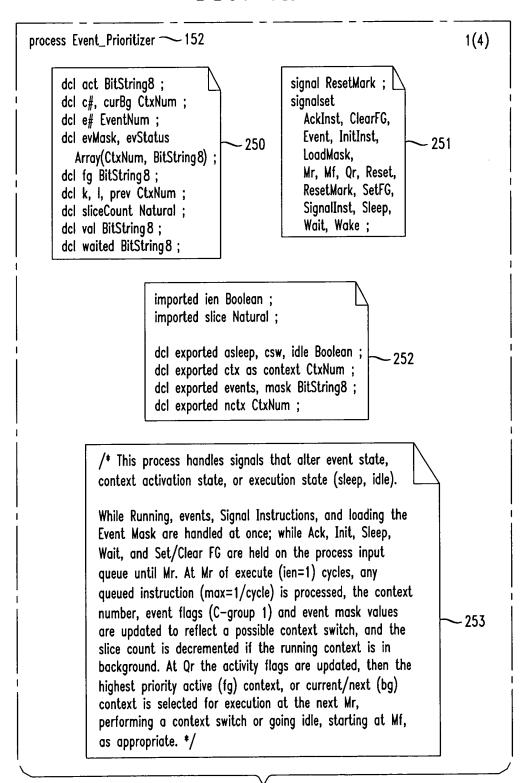Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

8/21

FIG. 7A1

process Event_Prioritizer ⁓152   1(4)

```
dcl act BitString8 ;
dcl c#, curBg CtxNum ;
dcl e# EventNum ;
dcl evMask, evStatus
    Array(CtxNum, BitString8) ;          ⁓250
dcl fg BitString8 ;
dcl k, l, prev CtxNum ;
dcl sliceCount Natural ;
dcl val BitString8 ;
dcl waited BitString8 ;
```

```
signal ResetMark ;
signalset
    AckInst, ClearFG,
    Event, InitInst,          ⁓251
    LoadMask,
    Mr, Mf, Qr, Reset,
    ResetMark, SetFG,
    SignalInst, Sleep,
    Wait, Wake ;
```

```
imported ien Boolean ;
imported slice Natural ;

dcl exported asleep, csw, idle Boolean ;      ⁓252
dcl exported ctx as context CtxNum ;
dcl exported events, mask BitString8 ;
dcl exported nctx CtxNum ;
```

```
/* This process handles signals that alter event state,
context activation state, or execution state (sleep, idle).

While Running, events, Signal Instructions, and loading the
Event Mask are handled at once; while Ack, Init, Sleep,
Wait, and Set/Clear FG are held on the process input
queue until Mr. At Mr of execute (ien=1) cycles, any
queued instruction (max=1/cycle) is processed, the context      ⁓253
number, event flags (C-group 1) and event mask values
are updated to reflect a possible context switch, and the
slice count is decremented if the running context is in
background. At Qr the activity flags are updated, then the
highest priority active (fg) context, or current/next (bg)
context is selected for execution at the next Mr,
performing a context switch or going idle, starting at Mf,
as appropriate. */
```

FIG. 7A2

FROM FIG. 7A1

254 — (rounded box)

init — 255

256 —
```
act := 0x80,
ctx := 7,
fg := 0xFF,
```

257 —
```
asleep := false,
csw := false,
idle := false
```

258 —
```
events := 0x00,
waited := 0x00,
mask := 0x00,
```

259 —
```
evMask(0,1,2,
3,4,5,6,7)
:= 0x00,
evStatus(0,1,2,
3,4,5,6,7)
:= 0x00
```

260 —
```
curBg := 0,
sliceCount :=
import(slice)
```

262
InitSeq(7)

264
ClearCy(7)

266 —
```
export
asleep, csw,
ctx, events,
```

267 —
```
idle,
mask,
nctx
```

268
```
CsLoad(7)
via PriSeq
```

270 — Running

272 — *

274 — Reset

276 — ResetMark to Self

278 — Flush_Signals

280 — ResetMark

init — 282

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
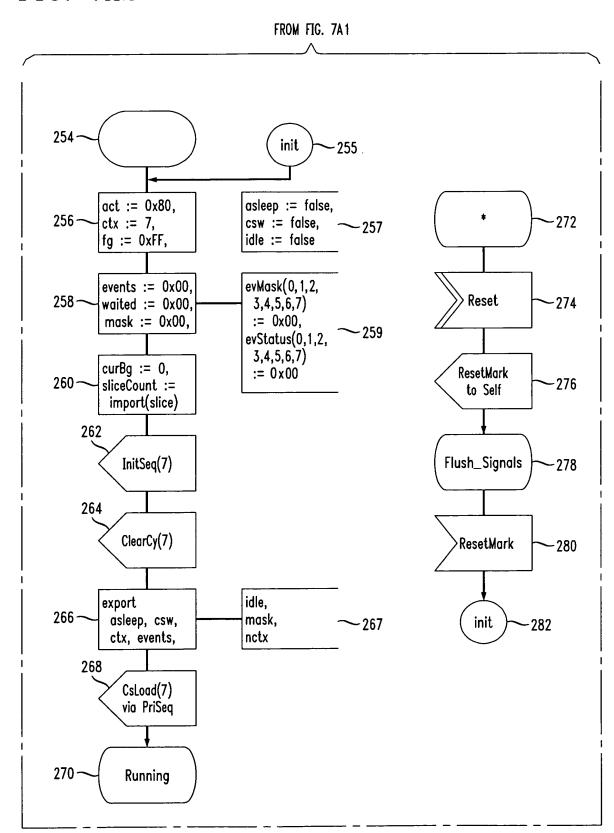Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

10/21

FIG. 7B1

TO FIG. 7B2

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
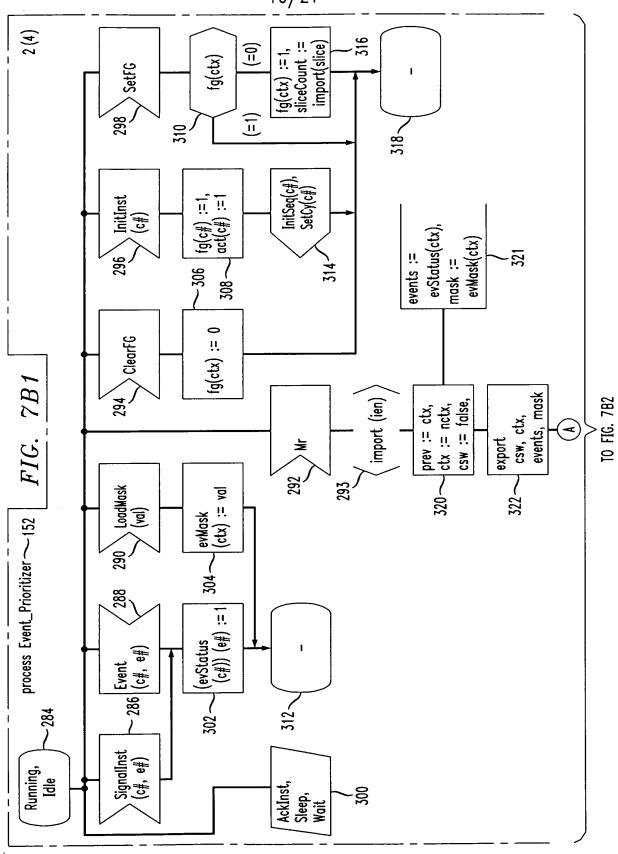Replacement Sheet

11/21

FIG. 7B2

FIG. 7C   12/21

process Event_Prioritizer —152   3(4)

Prioritize —350

AckInst (e#) —352

(evStatus(prev)) (e#) := 0 —354

Wait 360

waited(prev):= 1 362

364 —

Sleep 366

asleep := true 368

export asleep 370

Qr —380

AckEv (prev, e#) —356

waited(ctx) —382

(=0)   (=1)

act(ctx) := 0,
waited(ctx) := 0 —384

— —358

Sleep 372

— 374

c# := 0 —386

c# := c# + 1 —392

act(c# :=
if (evMask(c#)
and —388

evStatus(c#)) = 0
then act(c#)
else 1 fi —389

c# —390

else

(=7)

sel_
next —394

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
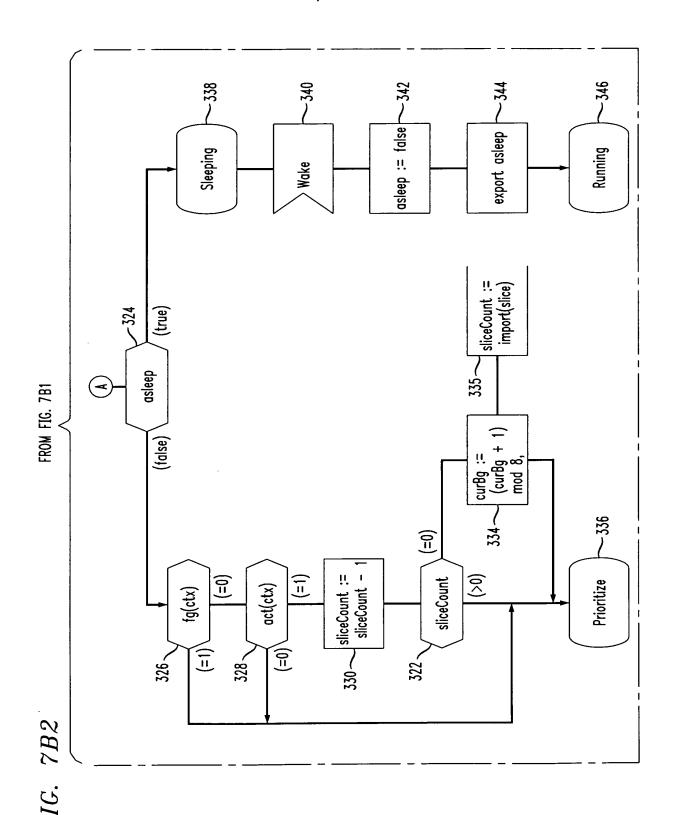Replacement Sheet

13/21

FIG. 7D1

process Event_Prioritizer

152

420 — sel_next

4(4)

424 — k := k + 1

422 — nctx := curBg, k := 0, l := if

423 — (act(ctx)=0) or fg(ctx)=0) then 7 else ctx fi

426 — fg(k) and act(k)

(=0)   (=1)

428 — k < 1

(true)

(false)

430 — not(fg(curBg)) and act(curBg)

(=0)   (=1)

450 — nctx := curBg

452 — nctx := k

432 — curBg := (curBg + 1) mod 8

434 — curBg=nctx

(false)

(true)

454 — ctx = nctx

(false)   (true)

436 — Go_Idle
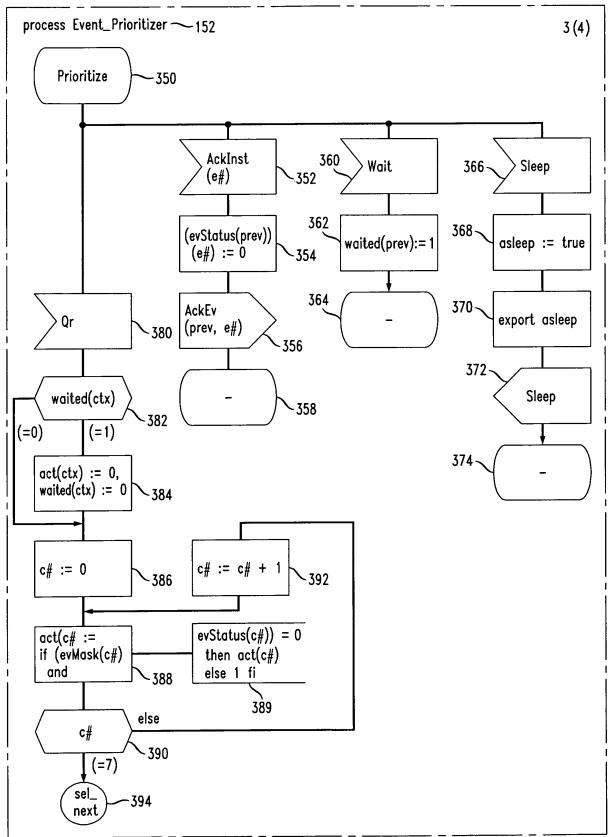
456 — Start_CSW

Running

458

438 — Mf

440 — *

462 — *

460 — Mf

(A)   (B)

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
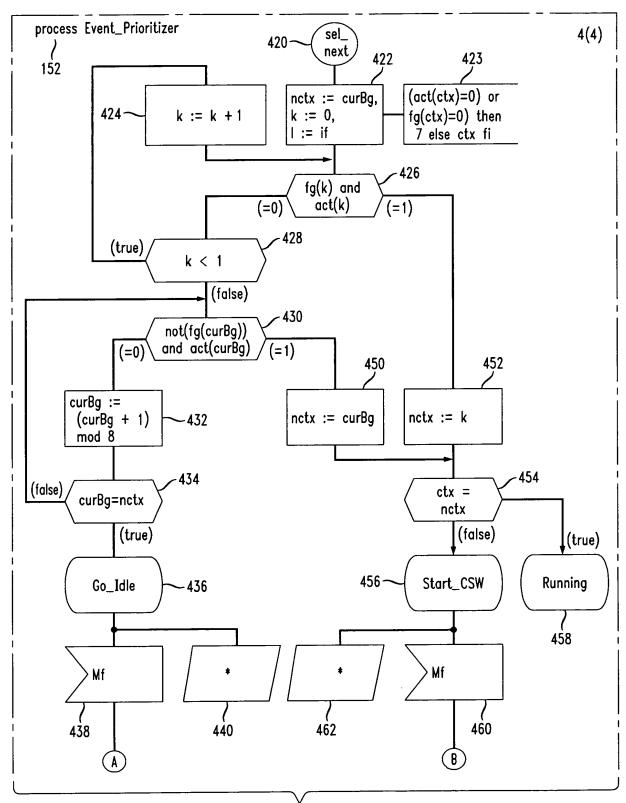Replacement Sheet

14/21

FIG. 7D2

FROM FIG. 7D1

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

15/21

FIG. 8

Execute Cycle (ien=true) with Context Switch — 500

Execute Cycle without Context Switch; or Stall, Stretch or Idle cycle (ien=false) — 502

Mr 517 · Qr 518 · Mf 519 · Qf 520

504 — MCLK

506 — QCLK

508 — CSW (522 indicates Context Switch; 524; 526)

510 — CTX[2:0]: Current Context Number (528) / Next Context Number (unchanged unless context switch) (530)

512 — NCTX[2:0]: Current Context Number (532) / Next Context Number (534) / Next Context Number (unchanged unless context switch) (536)

514 — Context Switch at Controller:

| 532 synchronize activation events at Mr | 533 prioritize & select next context | 534 new NCTX and CSW at Mf | 536 new CTX, negate CSW at Mr, first execute cycle of next context |

516 — Context Switch on Data Path:

| 538 start executing last instruction before switch, close source latches at Mf | 540 start read of next context's state at Mf | 542 synchronous write of old context state & destinations at Mr, start execution of next context |

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
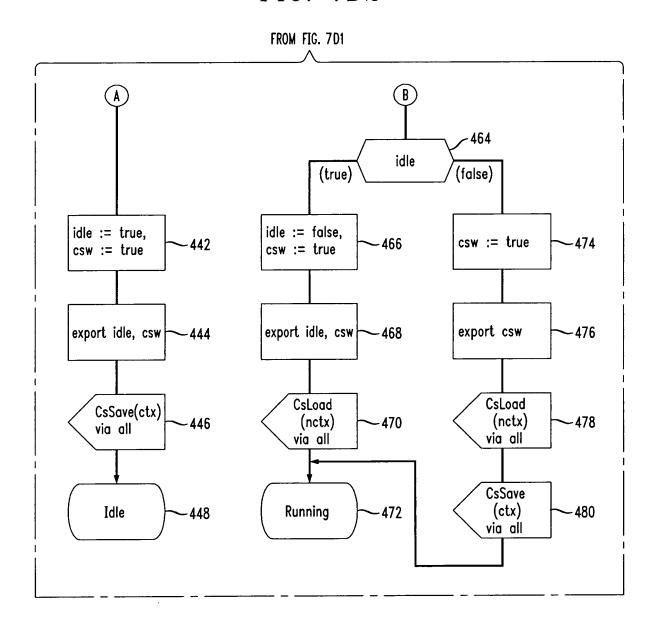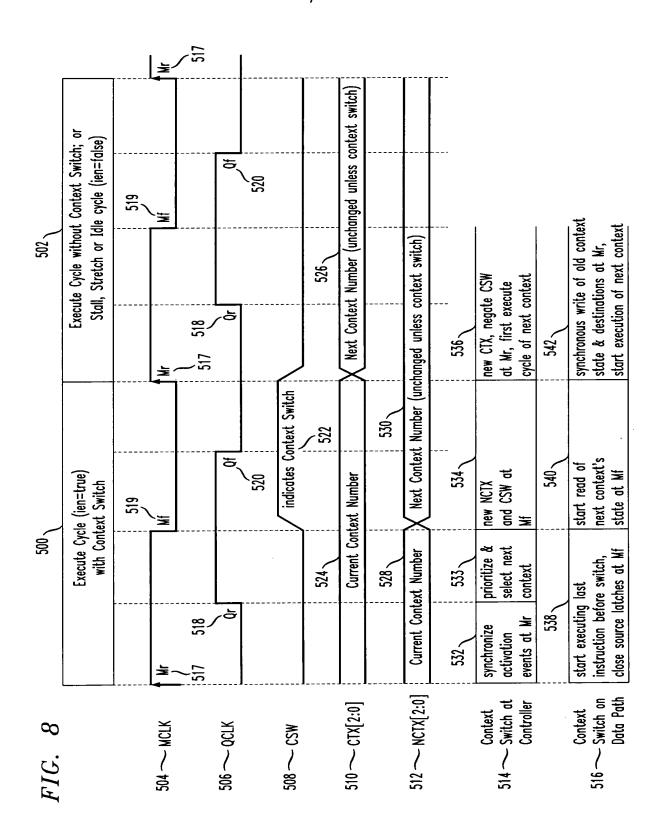Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
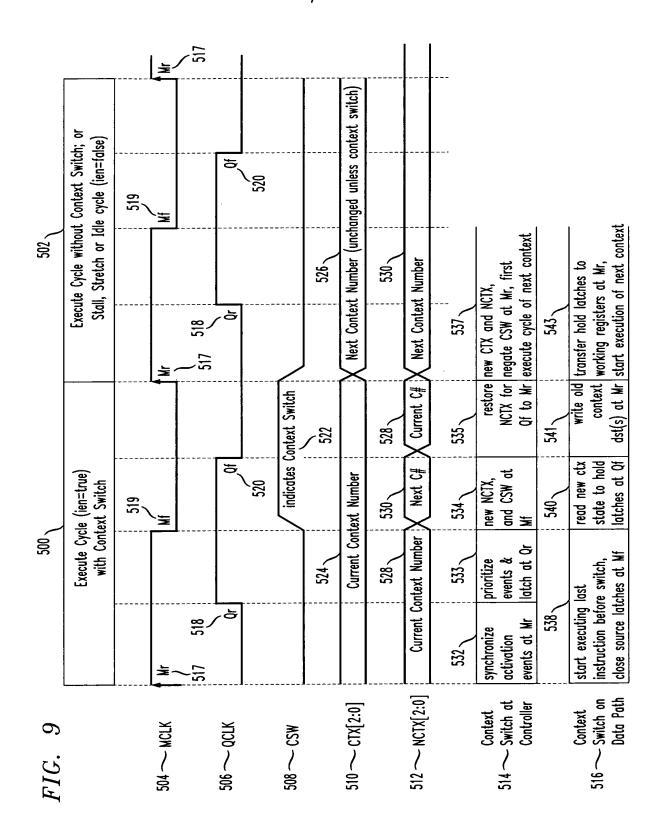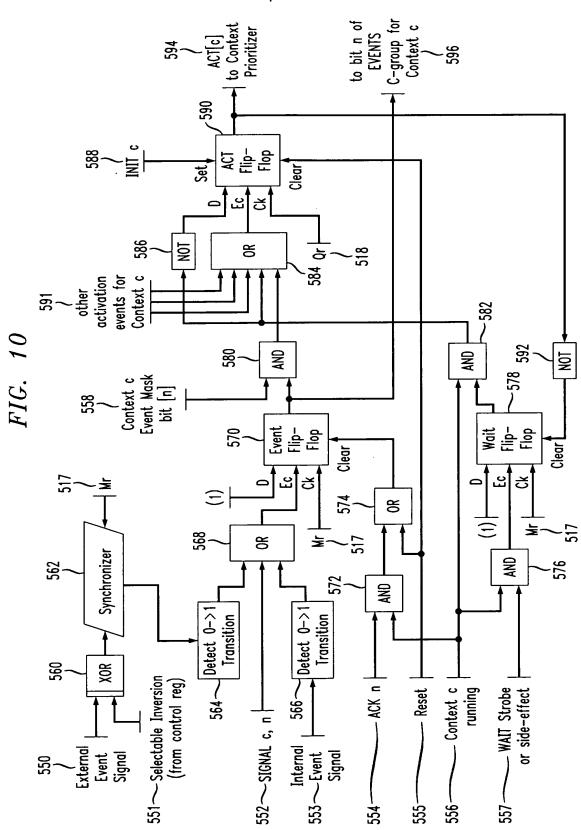Replacement Sheet

16/21

FIG. 9

FIG. 10

FIG. 11

SKPx 600

602	Test Operation 603	C-group 604	Mask Value: 605
0 ... 1	0 0 0	0 0	AFLAG – set flag values
	0 0 1	0 1	EVENTS – activation events (pre-Event Mask) ~608
	0 1 0	1 0	DBL0 – low-order byte of data bus
	0 1 1	1 1	DBH1 – high-order byte of data bus
	1 1 0		

SKP0 – skip if any bits = 1
SKP1 – skip if all bits = 0
SKP2 – skip if all bits = 1
SKP3 – skip if both 0s & 1s
SKP4 – skip if Cgroup=Mask

VECTOR 610

602	612	Vector Base (address bits 16:7) 613
0 ... 1	1 1 1	1 ... 1

SIGNAL 620

622	623	Event Number 624	Context Number 625

ACK 630

622	632	0 0	Event Number 624

INIT 640

622	642	0 1 0	not used	Context Number 625

STROBE 650

622	652	Control Function 653

					Control Function	
0	0	0	0	0	0	1
0	0	0	0	1	1	0
0	0	0	1	0	1	1
1	1	1	1	1	1	1

654 — WAKE – clear ACT bit
655 — SETFG – set FG bit
656 — CLRFG – clear FG bit
657 — SLEEP – begin sleep

FIG. 12

661 — control store accessed using even (word) addresses

	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
660 — Memory Address Bits																	
662 — Control Store Address					CSAddr[16:1] 664												0
666 — INIT Instruction	0	0	0	0	0	0	0	0	0	0	0	Context # 625			0	0	0
668 — VECTOR Instruction	0		Vector Base address [16:7] 613							Event # 624					0	0	0

DIEPSTRATEN 16-2-2
Context Controller Having Instruction-Based Time Slice Task Switching Capability and Processor Employing the Same
Attorney: J. Joel Justiss (972-480-8800)   Serial No. 09/213,970
Replacement Sheet

20/21

FIG. 13

Initialization Vectors	CS Word Addr ~678
670 Context 0 Initialization Vector	0000
671 Context 1 Initialization Vector	0004
672 Context 2 Initialization Vector	0008
673 Context 3 Initialization Vector	000C
674 Context 4 Initialization Vector	0010
675 Context 5 Initialization Vector	0014
676 Context 6 Initialization Vector	0018
677 Context 7 Initialization Vector	001C

FIG. 14

Vector Table in Control Store

690

Handler for vectored event 0	Base (addr=0 mod 64) —680
Handler for vectored event 1	Base + 8 words —681
Handler for vectored event 2	Base + 16 words —682
Handler for vectored event 3	Base + 24 words —683
Handler for vectored event 4	Base + 32 words —684
Handler for vectored event 5	Base + 40 words —685
Handler for vectored event 6	Base + 48 words —686
Handler for vectored event 7	Base + 56 words —687
	Base + 64 words —688

1-of-8 branch

Vector Base
(from instruction word)

613

address
bits 16:7

base ||
event
||000

692

address
bits 6:4

lowest, non-masked
Event Number
(from context controller)

694